УДК 378.147, 621.395.7

І. І. Пастернак
Національний університет "Львівська політехніка",
кафедра електронних обчислювальних машин

ПРИНЦИПИ ПРОЕКТУВАННЯ СОЦІАЛЬНОЇ МЕРЕЖІ З МІНІМАЛЬНИМ НАВАНТАЖЕННЯМ НА СЕРВЕРИ

Запропоновано та досліджено соціальну мережу з врахуванням мінімального навантаження на сервери. Показано принципи проектування соціальних мереж. Наведено результати тестування розробленої соціальної мережі та проаналізовано навантаження на сервери.

Ключові слова: соціальна мережа, клієнт, сервер.

PRINCIPLES OF SOCIAL NETWORK WITH MINIMAL LOAD ON SERVER

A social network and investigated with regard to minimum load on the server. The principles of designing social networks. The results of the tests developed by social network analysis and load on servers.

Key words: social network, client, server.

Вступ

Розглядаючи Internet з позицій суто технічних або організаційно-технологічних, неможливо дати точне визначення ні самої мережі, ні ступеня її впливу на сучасний світ. Мережа Internet сьогодні – це швидше соціальне явище. Вона перестала бути просто комп'ютерною мережею, формуючи не тільки абсолютно нові технології інформаційного обміну, але і принципово нові види бізнесу, освіти, методів пізнання навколишнього світу, формування світогляду. Саме соціальні аспекти розвитку Internet становлять сьогодні найбільший інтерес з погляду використання сучасних інформаційних технологій як у повсякденній діяльності людини, так і при побудові передових інформаційних систем управління різними соціально-економічними процесами в суспільстві.

Комп'ютерні мережі стали потужним засобом соціалізації та вирішення бізнесових, освітніх, комунікативних проблем суспільства. З їх появою виникла можливість нових форм спілкування та взаємодії – мережних, а також і проблема впорядкування такого спілкування, формування культурних та організаційних його засад як у просторі комп'ютерних мереж, так і поза ним.

З соціального погляду мережа – це група індивідуальних агентів, які розділять неформальні норми або цінності, крім тих, які необхідні для звичайних ринкових операцій. Соціальна мережа – соціальна структура, утворена індивідами або організаціями. Вона відображає зв'язки між ними через різноманітні соціальні взаємовідносини, починаючи з випадкових знайомств і закінчуючи тісними родинними зв'язками. Вперше термін запропонував в 1954 році Дж. А. Барнес (в роботі Classand Committeesin a Norwegian Island Parish, "HumanRelations") [1, 2].

Наведено проект соціальної мережі, який буде представляти собою ВЕБ-сервіс. На сайті розмістять оголошення про різні "події", що буде інтерактивно відображено на карті. Буде надано можливість додавати приватні "події", фотографії, відео, різні посилання, забезпечено синхронізацію з соціальними мережами, GoogleMaps і т.д. У майбутньому передбачено розроблення додатків для Android/iOs.

Аналіз останніх джерел та публікацій

Архітектура клієнт-сервер є домінуючою концепцію у створенні розподілених мережних застосувань і передбачає взаємодію та обмін даними між ними. Вона передбачає такі основні компоненти: набір серверів, які надають інформацію або інші послуги програмам, які звертаються до них; набір клієнтів, які використовують сервіси, що надаються серверами; мережа, яка забезпечує взаємодію між клієнтами та серверами. Сервери є незалежними один від одного. Клієнти також функціонують паралельно і незалежно один від одного. Немає жорсткої прив'язки клієнтів до серверів. Більш ніж типовою є ситуація, коли один сервер одночасно обробляє запити від різних клієнтів; з іншого боку, клієнт може звертатися то до одного сервера, то до іншого. Клієнти мають знати про доступні сервери, але можуть не мати жодного уявлення про існування інших клієнтів.

Загальноприйнятим є положення, що клієнти та сервери – це насамперед програмні модулі. Найчастіше вони знаходяться на різних комп'ютерах, але бувають ситуації, коли обидві програми – і клієнтська, і серверна, – фізично розміщуються на одній машині; в такій ситуації сервер часто називають локальним.

Типовим прикладом клієнт-серверної взаємодії є WWW. Існує величезна кількість ВЕБ-серверів, на яких розміщується та чи інша інформація. У найпростішому випадку ця інформація буде набором ВЕБ-сторінок, які можуть зберігатися на сервері у вигляді файлів, розмічених за допомогою мови розмітки HTML. Але ситуація переважно є складнішою; значна частина ВЕБ-ресурсів на сучасному етапі є динамічними, тобто вони не існують у заздалегідь підготовленому вигляді, а створюються безпосередньо в процесі обробки запиту від користувача.

Модель клієнт-серверної взаємодії визначається насамперед розподілом обов'язків між клієнтом та сервером. Логічно можна виокремити три рівні операцій: рівень представлення даних, який по суті є інтерфейсом користувача і відповідає за представлення даних користувачеві і введення від нього керівних команд; прикладний рівень, який реалізує основну логіку застосування і на якому здійснюється необхідна обробка інформації; рівень управління даними, який забезпечує зберігання даних та доступ до них. Дворівнева клієнт-серверна архітектура передбачає взаємодію двох програмних модулів: клієнтського та серверного. Залежно від того, як між ними розподіляються наведені вище функції, розрізняють: тонкий клієнт, в межах якого всю логіку застосування та управління даними зосереджено на сервері. Клієнтська програма забезпечує тільки функції рівня представлення; товстий клієнт, коли сервер тільки керує даними, а обробку інформації та інтерфейс користувача зосереджено на стороні клієнта [3]. Тонкими клієнтами часто також називають пристрої з обмеженою потужністю: кишенькові комп'ютери, мобільні телефони та ін.

Трирівнева клієнт-серверна архітектура передбачає відділення прикладного рівня від управління даними. Виокремлюється окремий програмний рівень, на якому зосереджується прикладна логіка застосування. Програми проміжного рівня можуть функціонувати під управлінням спеціальних серверів застосувань, але запускатися такі програми можуть і під управлінням звичайного ВЕБ-сервера. Нарешті, управляє даними сервер даних.

Найчастіше ВЕБ-сервер і серверні модулі проміжного рівня розміщуються на одному комп'ютері, будуть окремі і логічно незалежні програмні модулі.

На сучасному етапі для програмування модулів проміжного рівня використовують мову серверних сценаріїв PHP, а для управління даними – СУБД MySQL. Отже, зв'язку PHP-MySQL слід розглядати як стандартний інструмент для створення порівняно простих інтерактивних ВЕБ-сайтів та систем електронної комерції; близько 90 % комерційних систем сьогодні створюється саме на цій основі. Водночас як засоби управління даними, так і middleware-засоби можуть бути найрізноманітнішими. Так, для створення серверних застосувань, крім PHP, широко застосовують Java, Perl, Python. Взагалі технології створення розподілених, зокрема ВЕБ-додатків, стрімко розвиваються. Слід згадати про технології EJB (EnterpriseJavaBeans), CORBA, а також про .NET – порівняно нову ініціативу компанії Microsoft. Для зберігання даних та їх передавання часто використовують так звану розширену мову розмітки XML (ExtensibleMarkupLanguage).

У своєму розвитку технології “клієнт-сервер” пройшли кілька етапів, тому є різні моделі технології. Їх реалізація основана на поділі структури СКБД на три компоненти:

- введення і відображення даних (інтерфейс з користувачем);
- прикладний компонент (запити, події, правила, процедури та функції, які характерні для певної предметної області);
- функції керування ресурсами (файловою системою, базою даних тощо).

Тому в будь-якому додатку вирізняють такі компоненти:

- компонент подання даних;
- прикладний компонент;
- компонент керування ресурсом.

Зв’язок між компонентами відбувається за певними правилами, які називають “протокол взаємодії”. Існують різні класифікації, але однією з найпоширеніших є використання чотирьох моделей технології “клієнт-сервер”.

Постановка завдання

Запропонована соціальна мережа повинна забезпечити мінімальне навантаження на сервери та підтримувати одночасну роботу (кількість сесій) не менше ніж 2500 користувачів і витримувати короткочасне збільшення навантаження.

Основні результати досліджень

Архітектура проекту соціальної мережі з мінімальним навантаженням на сервери

Запит від користувача може надходити через ВЕБ-сайт або через додатки на мобільних пристроях. Можлива реєстрація через інші соціальні мережі. При цьому відбувається звернення до API соціальних мереж. Потім запит обробляється власною API на одному з Core серверів (рис. 1).

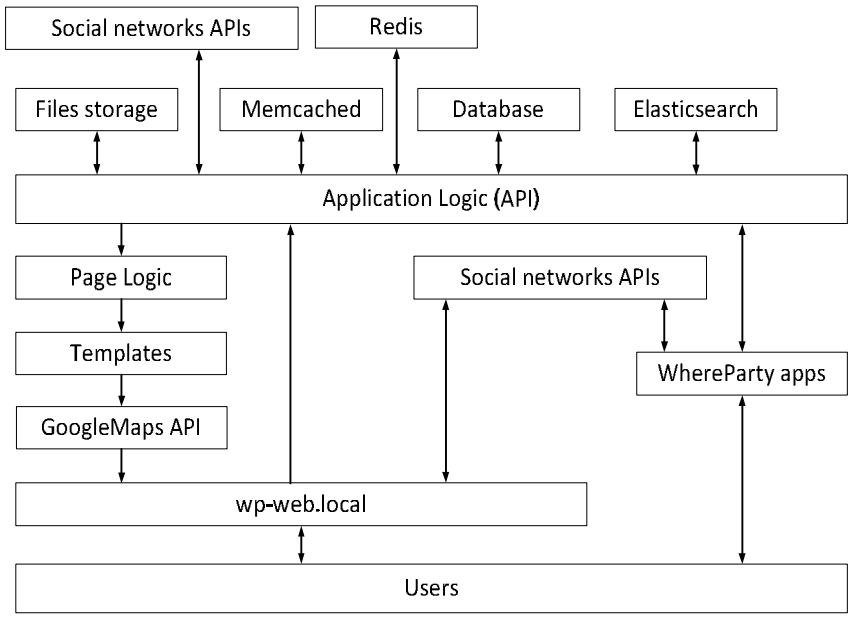

Рис. 1. Загальна архітектура соціальної мережі

З цього сервера звернення можуть надходити до Filestorage (медіафайли, фото та відео), Database(база даних), Elasticsearch (пошук в базі даних). Для кешування метаданих класів та запитів до бази даних використовується Memcached. Redis використовується для найбільш часто використовуваної інформації. Потім інформацію отримує користувач [4].

Для збільшення пропускної здатності каналу Б та зменшення кількості запитів на оновлення БД створено кластер. Як ORM використовували Doctrine2, яка надає надзвичайно гнучкі інструменти для налаштування БД та зв’язків. Для кешування зв’язків БД та метаданих об’єктів

використано інтегровані в Doctrine 2 методи кешування memcached. Також для кешування в оперативній пам'яті об'єктів, які найчастіше використовуються,та кешування сесій користувачів використовувався Redis. Для швидкого пошуку за БД та оптимізації застосовано Elasticsearch, який є дуже швидким і зручним методом розроблення високонавантажуваних проектів.

Архітектура серверної частини соціальної мережі

Отримавши запит від клієнта, ВЕБ-сервер передає його через LoadBalancer на Core Server або Redis Server для отримання сесії користувача. CoreServer також через LoadBalancer передає запит на один із серверів бази даних, об'єднаних у кластер або до одного з Redis серверів для отримання даних. За потреби CoreServer звертається до Elasticsearch Server під час пошуку (рис. 2).

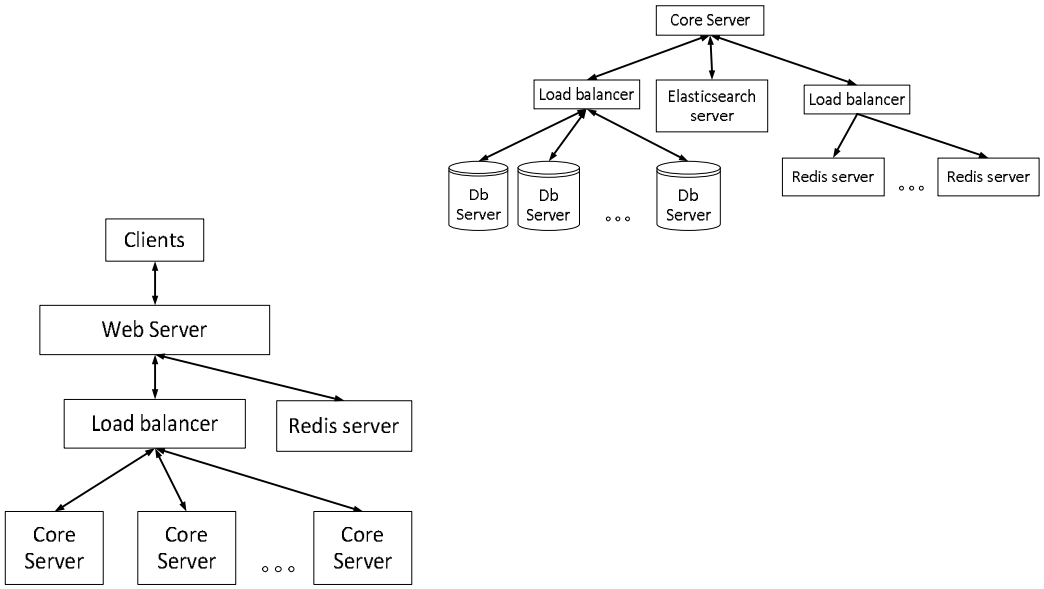

Рис. 2. Архітектура серверної частини соціальної мережі

Мінімального навантаження на сервери досягнуто завдяки правильній архітектурі серверів та оптимальній архітектурі БД. Для збільшення пропускної здатності каналу БД та зменшення кількості запитів на оновлення БД створено кластер, який складався з трьох серверів (Node) з можливістю подальшого горизонтального масштабування. Використовували технології, які часто застосовують для розроблення високонавантажуваних проектів. Як ORM використовували Doctrine2, яка надає надзвичайно гнучкі інструменти для налаштування БД та зв'язків. Для кешування зв'язків БД та метаданих об'єктів використана інтегровані в Doctrine 2 методи кешування memcached. Також для кешування в оперативній пам'яті об'єктів, які найчастіше використовуються, та кешування сесій користувачів використано Redis. Для швидкого пошуку по БД та оптимізації застосовано Elasticsearch, який є дуже швидким і зручним методом розроблення таких високонавантажуваних проектів, як соціальна мережа [5, 6].

Для віддаленого управління конфігураціями було використано Ansible. Ansible бере на себе всю роботу з приведення віддалених серверів у необхідний стан. Адміністратору необхідно лише описати, як досягти цього стану за допомогою так званих сценаріїв. Така технологія дає змогу дуже швидко переконфігуровувати систему: достатньо лише додати кілька нових рядків у сценарій. Переваги Ansible порівняно з іншими аналогічними рішеннями (тут насамперед слід назвати такі продукти, як Puppet, Chef і Salt) полягають у такому:

· на керовані вузли не потрібно встановлювати жодного додаткового ПЗ, все працює через SSH (у разі потреби додаткові модулі можна взяти з офіційного репозиторію);

· код програми, написаний на Python, дуже простий; за необхідності написання додаткових модулів не становить особливих труднощів;

· мова, якою пишуть сценарії, також проста;

· низький поріг входження: навчитися роботі з Ansible можна за дуже короткий час;

· документацію до продукту написано дуже докладно і разом з тим просто і зрозуміло; вона регулярно оновлюється;

· Ansible працює не тільки в режимі push, а й pull, як це роблять більшість систем управління (Puppet, Chef);

· мається можливість послідовного поновлення стану вузлів (rollingupdate).

Вимоги для установки Ansible мінімальні. На машині, з якої проводиться керування повинен бути встановлений Python 2.6 або вищий. На керованих вузлах повинен бути встановлений тільки Python версії, не нижчої 2.4, але він, як правило, за замовчуванням включений до складу більшості дистрибутивів linux-систем. MS Windows не підтримується.

Перед внесенням змін Ansible підключається до керованих вузлів і збирає інформацію про них: про мережеві інтерфейси та їхні стані, про встановлену операційну систему тощо. Він може робити це як за допомогою власного модуля, так і за допомогою інструментів ohai і facter, якщо їх встановлено.

До складу Ansible входить величезна кількість модулів для розгортання, контролю і управління різними компонентами, які можна умовно поділити на такі групи:

- хмарні ресурси і віртуалізація (Openstack, libvirt);
- бази даних (MySQL, Postgresql, Redis, Riak) [7];
- файли (шаблонізації, регулярні вирази, права доступу);
- моніторинг (Nagios, monit);
- оповіщення про виконання сценарію (Jabber, Irc, пошта, MQTT, Hipchat);
- мережа і мережева інфраструктура (Openstack, Arista);
- управління пакетами (apt, yum, rhn-channel, npm, pacman, pip, gem);
- система (LVM, Selinux, ZFS, cron, файлові системи, сервіси, модулі ядра);
- робота з різними утилітами (git, hg).

Структура інтерфейсу користувача соціальної мережі

На головній сторінці сайту є реєстрація або вхід зареєстрованого користувача. Зі сторінки користувача можна перейти на наступні сторінки:

– Налаштування (змінити пароль, налаштування приватності та інші налаштування інформації користувача);

– Події (Користувач може додавати або редагувати свої події та переглядати події інших користувачів);

– Друзі (приєднати друзів або видалити зі списку друзів, перевести в підписники, написати повідомлення);

– Повідомлення (користувач може вести, переглядати і видаляти діалоги, переглядати та створювати повідомлення);

– Групи (користувач може створювати та редагувати свої групи або переглядати списки груп);

– Пошук (можна вести пошук користувачів, груп, подій).

Для налаштування є 4 рівні приватності:

1. Інформація певного користувача публічна і доступна для будь-якого переглядача сторінки.
2. Інформація відкрита тільки для підписників.
3. Інформація відкрита для друзів.
4. Інформація приватна і закрита для перегляду будь-ким.

Під час пошуку або додавання події можна застосовувати фільтри. Різні події фільтруються за типом, місцем проходження, віковою категорією. Події можуть бути активні і вже закриті.

Для того, щоб почати роботу з інтерфейсом, необхідно відкрити вікно будь-якого встановленого браузера (Internet Explorer, Opera, Mozilla Firefox або інший) і у рядок адреси ввести адресу ВЕБ-сайту [8].

При вході в систему користувач бачить перед собою форму для швидкої реєстрації (рис. 3). Користувач може ввести дані в поля введення і увійти в систему під новим акаунтом.

Рис. 3. Структура інтерфейсу користувача соціальної мережі

На електронну пошту користувача буде відправлено лист для підтвердження e-mail. Для входу на сайт потрібно вказати свою електронну адресу, логін та пароль.

Якщо користувач забув свій пароль, передбачено процедуру відновлення з відправленням пароля на електронну адресу.

За допомогою GoogleMaps користувач може позначати на карті місце події, яка відбуватиметься. При пошуку користувачем подій, які його цікавлять, система за IP-адресою визначає місцерозташування користувача і видає насамперед події, які відбуваються в районі його розташування (рис. 3).

Також на сайті є можливість додавати друзів, переглядати події, на які вони запрошують або шукають. На сторінці, що представлена на скріншоті, зображено іконки всіх друзів певного користувача соціальної мережі.

Розглянемо структуру бази даних проекту соціальної мережі з мінімальним навантаженням на сервери (рис. 4).

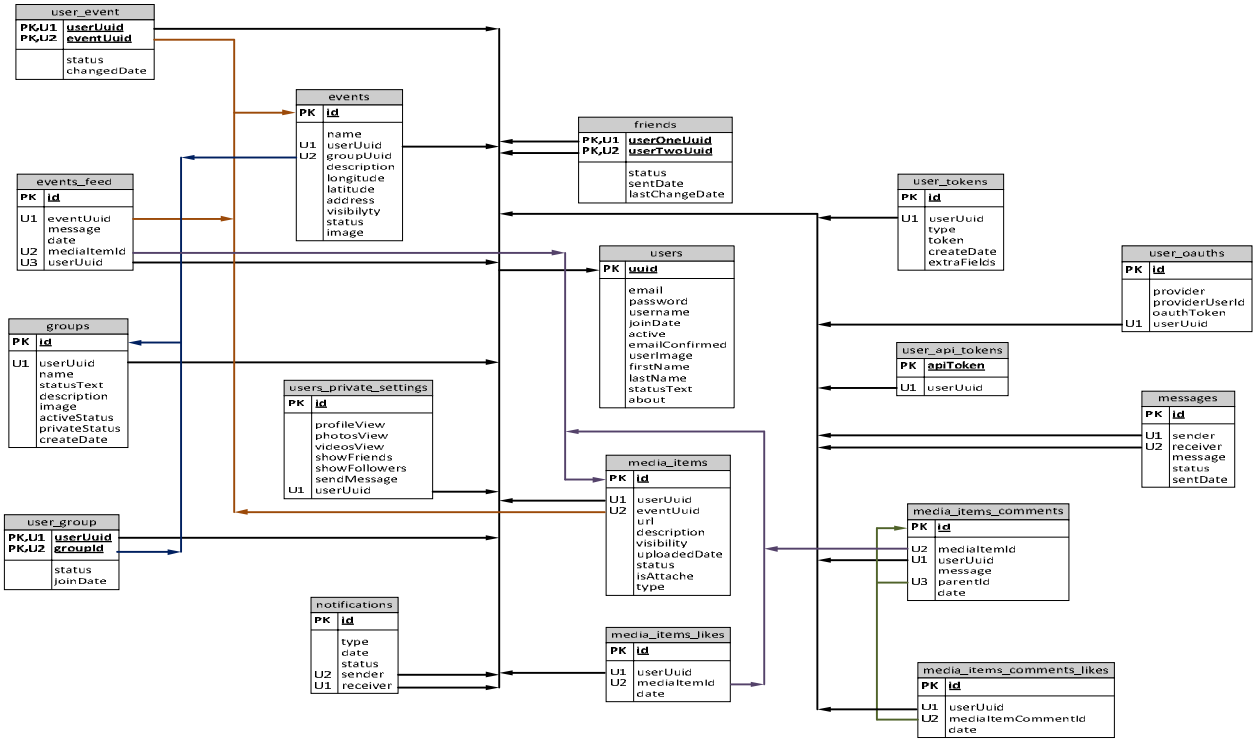

Рис. 4. Структура таблиць бази даних соціальної мережі

Структура бази даних соціальної мережі, наведена на рис. 4, містить 17 таблиць:

1. Users – містить список всіх користувачів соціальної мережі та інформацію про них.
2. Friends – відображає відношення між користувачами.
3. Groups – список груп, які може створити User.
4. Events – список подій, які створює User.
5. User_events – відображає зв'язок між Users та подіями.
6. Events_feed – записи та фото на "стіні".
7. User_group - відображає зв'язок між Users та групами.
8. User_private_settings – налаштування приватності користувача.
9. Notification – швидкі повідомлення.
10. Media_items – розміщені фотографії та відео.
11. Media_items_likes – фотографії та відео, які отримали likes.
12. User_tokens – зашифрований код для відновлення пароля.
13. User_api_tokens – система аутентифікації users через ApiTokens.
14. Media_items_comments – коментарі до фото та відео.
15. Media_items_comments_likes- коментарі до фото та відео, які отримали likes.
16. User_oauths – залогінення через іншу мережу.
17. Messages – список повідомлень.

Всі таблиці пов'язані між собою різними типами зв'язків.

Для впровадження проекту соціальної мережі використовують виділені сервери, які витримують більше навантаження, ніж віртуальні.

У проекті передбачено залучити

3 Coreсервера;

3 Db сервера;

1 ВЕБ сервер;

2 Redis сервера;

1 Redis сервер для ВЕБ;

1 Load Balancer сервер (ядро);

1 LoadBalancerсервер (база даних);

1 LoadBalancerсервер (Redis сервер);

CI сервер (Continuous integration).

При використанні 1 сервера бази даних і одного Coreсервера кількість користувачів в системі приблизно 800, а збільшення кількості серверів до 3-х збільшує кількість клієнтів до 3200. При цьому навантаження на сервери знижується приблизно на 15 % завдяки обраній архітектурі серверів та оптимальній архітектурі БД.

Висновки

У результаті проектування соціальної мережі було розроблено робочу версію інтерфейсу користувача соціальної мережі з мінімальним навантаженням на сервери. Реалізовано базову функціональність. Для відображення подій, які можуть зацікавити користувача, додано карти Google, організовано синхронізацію з соціальними мережами, фото-, відеогалерею, гнучкий пошук за користувачами та подіями. Принциповою особливістю вищенаведеної соціальної мережі є орієнтація на відкрите програмне забезпечення. Програмували модулі системи мовою програмування PHP з використанням платформи web-розробки Symfony компанії Sensio Labs. Як СУБД було обрано систему MySQL 5.2. Розроблено та реалізовано мережну архітектура на основі ВЕБ-сервера Nging.

У розробленому проекті соціальної мережі мінімального навантаження на сервери досягнуто завдяки обраній архітектурі серверів та оптимальної архітектури БД. Для збільшення пропускної здатності каналу БД та зменшення кількості запитів на оновлення БД створено кластер, з 3 серверів

(Node) з можливістю подальшого горизонтального масштабування. Використовували технології, які часто застосовують для розроблення високонавантажуваних проектів. Як ORM використовувалась Doctrine2, яка надає надзвичайно гнучкі інструменти для налаштування БД та зв'язків. Для кешування зв'язків БД та метаданих об'єктів використано інтегровані в Doctrine 2 методи кешування memcached. Також для кешування в оперативній пам'яті об'єктів, які найчастіше використовуються, та кешування сесій користувачів використано Redis. Для швидкого пошуку по БД та оптимізації застосовано Elasticsearch, який є дуже швидким і зручним методом розроблення високонавантажуваних проектів.

1. Астахова И. Ф. SQL в примерах и задачах : учеб. пособ. / И. Ф. Астахова, А. П. Толстоборов, В. М Мельников. – Минск : Новое знание, 2007. – 160 с. 2. Бабенко В. Г., Шадхін В. Ю., Компанієць В. О. Оперативний розподіл навантаження на мережі передачі даних / Вісник Хмельницького національного університету. – 2010. – № 3. – С. 23–28. 3. Березко Л. О., Якимець А. І. Ефективний метод обробки запитів до ВЕБ-сервісів // Вісник Нац. ун-ту "Львівська політехніка" "Комп'ютерні системи та мережі". – 2014. – № 745. – С. 12–17. 4. Виейра Роберт. Программирование баз данных Microsoft SQL Server 2005. Базовый курс.: Пер. с англ. – М.: ООО "Вильямс", 2005. – 832с. 5. Грабер М. Справочное руководство по SQL. – М.: Лори, 2003. 293 с. 6. Жуков І. А. Методи балансування навантаження для Web-серверів / І. А. Жуков, І. М. Кравець // Проблеми інформатизації та управління.: Збірник наукових праць. – К. НАУ, 2007. – С. 120–124. 7. Моисеев Т. Н., Распределение информационных потоков данных в распределенных многосерверных системах / Т. Н. Моисеев. – Воронеж: Научная книга, 1998. – 140 с. 8. Нильсен Пол. MS SQL Server 2005. Библия пользователя: Пер. с англ. – М.: ООО "Вильямс", 2005. – 238 с.